TCP Variants Performance Analysis in Mobile Ad Hoc Networks

Rajkiran K[#1], Rakshith P Ram[#2], Siddlingappagouda C Biradar[#3]

[#1]Student, Department of Electronics and Communication Engineering, Don Bosco Institute of technology, Bangalore – 560074, Karnataka, India.

[#2]Student, Department of Electronics and Communication Engineering, Don Bosco Institute of technology, Bangalore – 560074, Karnataka, India.

[#3]Assistant Professor, Department of Electronics and Communication Engineering, Don Bosco Institute of technology, Bangalore – 560074, Karnataka, India.

***Abstract*—In wired network the dominating protocol which is used for secured and reliable data transfer is nothing but the transport layer protocol TCP (Transmission Control Protocol) together with some other protocol in the protocol stack. Its performance is best in wired networks but if the same protocol is used in wireless networks then the performance reduces. In this paper performance of the different TCP agents are analyzed through network simulation for wireless network. Results from simulation are used to conclude which is the better clone in wireless mobile ad hoc networks.**

Keywords-TCP, Newreno, Reno, Vegas, Mobile Ad Hoc Network, NS2.

I. INTRODUCTION

In the present days the web packet data services for applications such as e-mail, file transfer, web browsing etc. are increasing highly. And as a result TCP [1] which is the end to end routing protocol on the internet presently carrying more than 92 percentage of the total packet traffic. It provides a proper secure and reliable wireless or wired connection between two sources in a multi-network scenario. TCP agent appears in different clones (such as Newreno, Reno, and Vegas etc.). All are with different advantages, but with maximum throughput as its main objective. The performance of a protocol for different network environment and topology can effectively be evaluated through simulation.

It has been analyzed that these TCP agents perform better in traditional wired networks where as packet losses are mostly caused by wireless network congestion. But the same clones perform differently when applied to wireless network where packet losses are due to different factors like movement of nodes or delay in the acknowledgement due to motion etc. It considers each packet loss as a part of congestion and aggressively initiates the congestion control algorithm which resulting in decrease in the link utilization and results into a significant degradation in the performance in the form of poor throughput. Different approaches are there to handle these situations are present and are differentiated into three different types [2] and some of them are worked in [3].

In this paper we have analyzed three different types of TCP agents named Newreno, Reno, and Vegas. We have compared and analyzed the results of these clones so as to find out which is the best of these clones in wireless network. The rest of the paper is divided as below. Section II provides a work that is related recent research work carried out. Section III gives the simulation scenarios. Simulation results and comparative analysis of different techniques are described in section IV. Section V provides the conclusion of this paper. Future enhancements are mentioned in the last section.

II. RELATED WORK

A. TCP Reno

This TCP Reno is built with the basic principle of TCP Tahoe, such as start with slow and the coarse grain retransmit timer. However it adds some intelligence over it so that lost packets are identify earlier and the pipeline is not emptied every time a packet is lost. Reno requires that we receive immediate acknowledgement whenever a segment is received. The logic behind this is that whenever we receive a duplicate acknowledgment packet, then his duplicate acknowledgment packets could have been received if the next segment in sequence expected, has been delayed in the network and the segments reached there out of order or else that the packet is lost. If we receive a number of duplicate acknowledgement packet then that means that sufficient time have passed and even if the segment had taken a longer path, it should have gotten to the receiver by now. There is a very high probability that it was lost. So Reno suggests that an algorithm called 'Fast Re-Transmit'. Whenever we receive three duplicate ACK's packets we take it as a sign that the segment was lost, so we re-transmit the segment without waiting for timeout.

Thus it manages to re-transmit the segment with the pipe almost full. Another modification that reno makes is in that after a packet loss, it does not reduce the congestion window to 1. Since this empties the pipe. It enters into an algorithm which we call **'Fast-Re-Transmit'.**

B. TCP Newreno

New reno is a quite change over tcp-reno. It is able to detect many number of packet losses and as a result it is slightly better that reno in the event of more packet losses. Like reno,

New-reno also enters into fast-retransmit when it gets many duplicate packets, however it different from tcp reno in which it doesn‘t exit fast-recovery until all the data packet which was out standing at the time it entered fast recovery is received. The fast-recovery phase proceeds as in Reno, however when a few ACK packets are received then we have two types of cases:

1)If it ACK‘s all the segments which were outstanding when we entered fast recovery then it exits fast recovery and sets CWD to threshold value and continues congestion avoidance like Tahoe.

2) If the ACK is a partial ACK then it deduces that the next segment which is in the line was lost and it re-transmits that segment and sets the number of duplicate ACKS received to zero. It exits Fast recovery when all the data in the window is acknowledged.

C. TCP Vegas

TCP Vegas is one among TCP implementation which is a modified version of TCP reno. It built based on the fact that proactive measure to encounter congestion is much more efficient than reactive protocol. It tried to get around the problem of coarse grain timeouts by suggesting an algorithm which checks for timeouts at a very efficient schedule. Also it overcomes the problem of duplicate acknowledgements packets to detect a packet loss, and it also suggests a change slow start algorithm which avoids from congesting the wireless network. The three major changes induced by TCP Vegas are:

New Re-Transmission Mechanism*:* TCP Vegas extend on the re-transmission mechanism of reno. It keeps track of when each segment was sent and it also calculates an estimate of the RTT by keeping track of how long it takes for the acknowledgment to get back.

Congestion avoidance: TCP Vegas is different from all the other implementation in its behavior during congestion avoidance. It does not use the loss of segment to signal that there is congestion. It determines congestion by a decrease in sending rate as compared to the expected rate, as result of large queues building up in the routers. It uses a variation of Wang and crow croft‘s Tri-S scheme.

Modified Slow-start: TCP Vegas differs from the other algorithms during its slow-start phase. The reason for this modification is that when a connection first starts it has no idea of the available bandwidth and it is possible that during exponential increase it over shoots the bandwidth by a big amount and thus induces congestion. To this end Vegas increases exponentially only every other RTT, between that it calculates the actual sending through put to the expected and when the difference goes above a certain threshold it exits slow start and enters the congestion avoidance phase.

III. Simulation Environment

We have used the open source discrete event network simulator ns-2[8], in order to simulate the various TCP variants. The latest version used for creating a wireless network is ns allinone-2.35. In simulation environment study we have used around 10 nodes and simulation time is 150 seconds. Nodes were arranged in a 200 X 200 mm grid. We have used Omni-directional antenna and AODV routing protocol [9].

IV. Simulation Results And Comparison

A. Number of Mobile Nodes vs Throughput

We have measured the throughput by changing the number of mobile nodes. The goodput parameter is considered actual throughput of the link. The goodput is the application level throughput, i.e. the number of useful information bits, delivered by the network to a certain destination, per unit of time. The amount of data considered excludes protocol overhead bits as well as retransmitted data packets. This is related to the amount of time from the first bit of the first packet is sent (or delivered) until the last bit of the last packet is delivered.

For example, if a file is transferred, the goodput that the user experiences corresponds to the file size in bits divided by the file transfer time. The goodput is always lower than the throughput (the gross bit rate that is transferred physically), which generally is lower than network access connection speed (the channel capacity or bandwidth).

Fig. 1 shows the throughput measured for all the three clones. And it is observed that Newreno has the greater throughput than others. Throughput also depends on various simulation factors like how fast the node moves (here we have taken 15m/s) and where is the position of the node after movement.

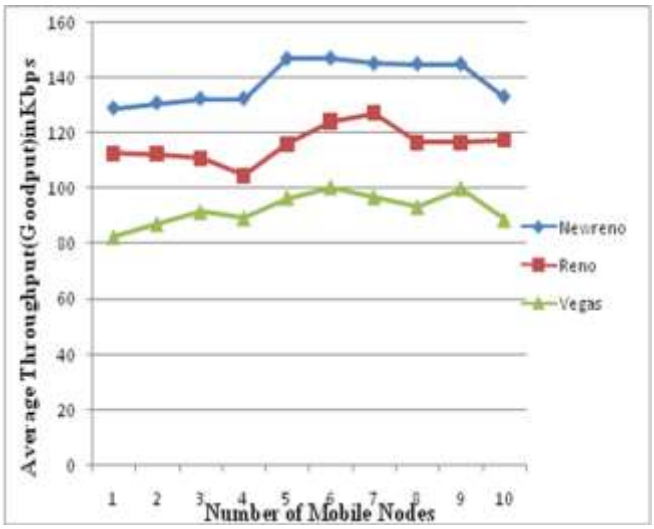

Fig. 1. Number of Mobile Nodes vs. Throughput

B. Number of Mobile Nodes vs End-to-End Delay

The average end-to-end delay of data packets is the interval between the data packet generation time and the time when the last bit arrives at the destination.

We have measured the end-to-end delay in the network through simulation by changing the number of mobile nodes. As shown in the Fig. 2 Newreno has the highest end-to-end delay compared to others. Vegas is the best one which takes less end-to-end delay.

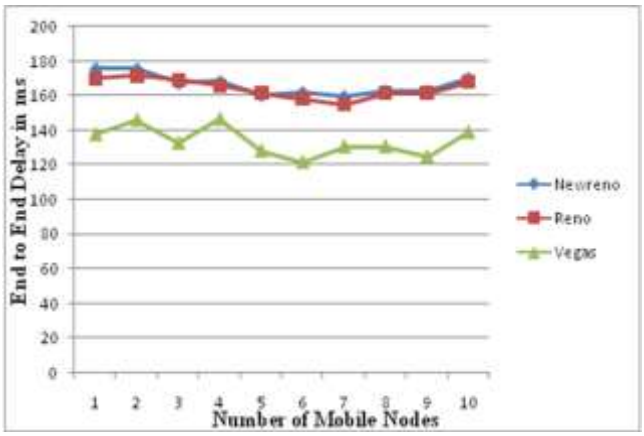

Fig. 2. Number of Mobile Nodes vs. End-to-End Delay

C. Number of Mobile Nodes vs Packet Delivery Ratio

The ratio of the number of packets originated by the application layer sources to the number of packets successfully delivered to their sink at the final destination. s From the Fig. 3 it is clear that the performance of Vegas is best among the three other clones of TCP. It is very important factor since it decides that even in large networks how fast the packets are delivered to the destination from the source.

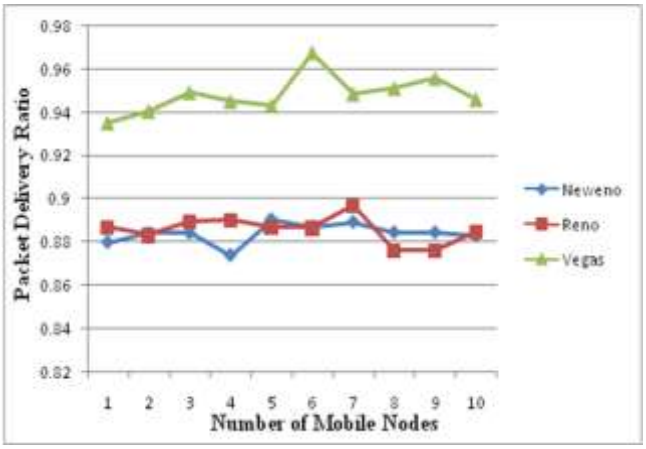

Fig. 3. Number of Mobile Nodes vs. Packet Delivery Ratio

V. Concusion And Future Enhancements

We all know that nowadays wireless communication plays important role in the field of communication. So it is important to make the wireless communication and its technology in a better stage. Many researches activities are going on this field. We found the performance of three types of TCP agent variants; they are TCP Reno, TCP Newreno and TCP Vegas. After comparing and analyzing the performance from simulated data and its graphs is obtained, in which we found that TCP Vegas is better than the other two TCP agents for sending data packets and information due to its better Packet Delivery Fraction and Average. end-to- end delay. This is due to fine tuning of congestion window size by taking into consideration the RTT of a packet, whereas other reactive protocols like TCP Reno and Newreno continue to increase their window size until packet loss is detected. We have given the detailed behavior of all these variants of TCP under various different scenarios. We can come to the conclusion that the simulation results will be of some use in future research study in this area helping the growing interest and resulting in the required protocol for today‘s high demanding world.

As a future work of this project we are currently working on these following areas:

1) Increasing the type of analysis by considering other new TCP agents like TCP Tahoe, TCP Sack, TCP Fack, etc

2) Considering more performance metrics like Routing Overhead, Total number of dropped packets etc.

3) A new TCP variant can be proposed which overcomes the drawbacks of other TCP clones.

References

[1] J. Postel, “Transmission Control Protocol,” RFC 793, September 1981.

[2] H. Balakrishnan, V. N. Padmanabhan, S. Seshan and R. H. Katz, “A Comparison of Mechanisms for Improving TCP Performance over Wireless Links”, IEEE Transactions on Networking, vol. 5, no. 6, pp. 756 – 769, December 1997.

[3] A. Natani, J. Jakilinki, M. Mohsin, and V. Sharma, “TCP for Wireless Networks”, SIGCOMM, vol. 26, no. 5, pp. 163–243, 2001.

[4] Md. Mohsin Ali, A. K. M. Sazzadul Alam, and Md. Shohan Sarker “TCP Performance Enhancement in Wireless Mobile Ad Hoc Networks”, International Journal on Internet and Distributed Computing Systems. Vol: 1 No: 1, 2011.

[5] Yuvaraju B. N, Niranjan N Chiplunkar “Scenario Based Performance Analysis of Variants of TCP using NS2”, International Journal of Advancements in Technology ,Vol 1, No 2.

[6] R. Manoharan and E. Ilavarasan “Impact of Mobility on the Performance of Multicast Routing Protocols in MANET”, International Journal of Wireless & Mobile Networks , Vol.2, No.2, May 2010.

[7] A.Boomarani Malany, V.R.Sarma Dhulipala, and M.Chandrasekaran "Throughput and Delay Comparison of MANET Routing Protocols", Int. J. Open Problems Compt. Math., Vol. 2, No. 3, September 2009 ISSN 1998-6262, 2009 www.icsrs.org.

[8] "Network Simulator", http://www.isi.edu/nsnam/ns/.

[9] P. Meeneghan and D. Delaney, "An Introduction to NS, Nam and OTcl Scripting,